PhishTime:
Continuous Longitudinal Measurement of the Effectiveness of Anti-phishing Blacklists

Adam Oest[*], Yeganeh Safaei[*], Penghui Zhang[*]
Brad Wardman[†], Kevin Tyers[†], Yan Shoshitaishvili[*], Adam Doupé[*], Gail-Joon Ahn[*,§]

[*]Arizona State University, [†]PayPal, Inc., [§]Samsung Research
[*]{aoest, ysafaeis, pzhang57,yans, doupe, gahn}@asu.edu, [†]{bwardman, ktyers}@paypal.com

Abstract

Due to their ubiquity in modern web browsers, anti-phishing blacklists are a key defense against large-scale phishing attacks. However, sophistication in phishing websites—such as evasion techniques that seek to defeat these blacklists—continues to grow. Yet, the effectiveness of blacklists against evasive websites is difficult to measure, and there have been no methodical efforts to make and track such measurements, at the ecosystem level, over time.

We propose a framework for continuously identifying unmitigated phishing websites in the wild, replicating key aspects of their configuration in a controlled setting, and generating longitudinal experiments to measure the ecosystem's protection. In six experiment deployments over nine months, we systematically launch and report 2,862 new (innocuous) phishing websites to evaluate the performance (*speed* and *coverage*) and consistency of blacklists, with the goal of improving them.

We show that methodical long-term empirical measurements are an effective strategy for proactively detecting weaknesses in the anti-phishing ecosystem. Through our experiments, we identify and disclose several such weaknesses, including a class of *behavior-based* JavaScript evasion that blacklists were unable to detect. We find that enhanced protections on mobile devices and the expansion of evidence-based reporting protocols are critical ecosystem improvements that could better protect users against modern phishing attacks, which routinely seek to evade detection infrastructure.

1 Introduction

Phishing attacks represent a significant threat to millions of Internet users [62]. Beyond stealing victims' account credentials, modern phishing websites have evolved to collect extensive financial and personal information to fuel identify theft, fraud, and other cybercrime [29, 57]. Simultaneously, phishing inflicts collateral damage by harming the reputation of impersonated brands, compromising legitimate infrastructure, and necessitating effort to mitigate abuse [45].

The anti-phishing ecosystem has long been involved in a cat-and-mouse game with attackers (phishers). Despite the ecosystem's evolving defenses, the volume of phishing websites has continued to grow over time and has recently reached record-high levels [2, 25]. Phishing remains popular among criminals due to its scalability and low barrier to entry—even for sophisticated and highly evasive attacks—thanks to the support of illicit underground services [9, 28].

Robust yet scalable ecosystem-level defenses are thus needed to protect users from the modern barrage of phishing. Anti-phishing blacklists, which alert users whenever they try to visit a known malicious website, and are enabled by default in major desktop and mobile web browsers, are a key defense [52]. Blacklists are supported by extensive backend infrastructure that seeks to detect and mitigate phishing attacks.

Despite the importance of blacklists, and even attention from security researchers [44, 50, 52, 63], there have been no systematic, long-term, real-world studies of the anti-phishing blacklist ecosystem. Evasive phishing attacks that attempt to circumvent blacklists are not only becoming more common, but have recently been shown to be responsible for the majority of real-world impact due to large-scale phishing [46]. Thus, the blacklisting of such attacks warrants close scrutiny.

In this paper, we propose PhishTime: a framework for continuously identifying sophisticated phishing attacks in the wild *and* continuously monitoring—in an empirical, controlled manner—the response of the anti-phishing ecosystem to blacklist evasion techniques, with the goal of automatically identifying gaps within the ecosystem. PhishTime can thus be used to ensure that the ecosystem—or specific entities within it—deliver a consistent degree of protection to users. In the first longitudinal study of its kind, we deploy the framework over the course of one year to measure the performance of three blacklists—*Google Safe Browsing*, *Microsoft SmartScreen*, and *Opera*—across major desktop and mobile browsers (which collectively have an overwhelming global market share [8]).

PhishTime operates in two stages: first, it collects phishing URL reports in real time and monitors the blacklisting status of *live* phishing websites (run by actual criminals). Criminal phishing websites that evade prompt blacklisting are manually analyzed for insight into evasion techniques successful against blacklists. Second, PhishTime leverages these insights to generate experiments that deploy large batches of artificial (realistic, yet innocuous) phishing websites with evasion techniques representative of those observed in the criminal

websites. Then, PhishTime adapts and enhances a previously proposed, automated testbed [44] to host the artificial phishing websites, report them to blacklists, and measure the blacklists' response (while implementing numerous controls to minimize confounding factors). Unlike prior empirical studies, PhishTime's experimental methodology uniquely enables it to evaluate and contextualize the response time of blacklists [44].

Our experiments involved the deployment, reporting (to blacklists), and monitoring of 2,862 new, previously unseen, artificial, evasive PayPal-branded phishing websites over a period of nine months. This yielded several interesting findings, which we promptly disclosed to the affected entities.

1. Blacklists exhibited an average response time of as little as 55 minutes against unsophisticated phishing websites, but phishing websites with evasion commonly used in the wild—even trivial techniques such as redirection via URL shorteners—delayed blacklisting up to an average of 2 hours and 58 minutes[1], and were up to 19% less likely to be detected. We also found that blacklists allow phishers to reuse domains for multiple attacks: with evasion, phishing websites reusing domains were still blacklisted up to 1 hour and 20 minutes slower than unevasive ones. Moreover, certain sophisticated JavaScript evasion could *entirely* avoid blacklisting.
2. PhishTime's continuous measurements enabled us to identify emerging issues over time. We detected a decrease in blacklisting seemingly due to a failure in PayPal's crawler-based phishing detection system (this finding led directly to remediation of this issue by PayPal). We also found a regression in the blocking of malicious redirections by *bit.ly* (but, unfortunately, received no response from that company). Lastly, mobile Chrome, Safari, and Opera consistently exhibited a lesser degree of blacklisting than their desktop counterparts.
3. New *evidence-based* phishing reporting protocols (i.e., that allow the submission of evidence such as a screenshot [11]) can expedite the blacklisting of evasive phishing websites. We perform the first comparison of such a protocol alongside traditional URL-only reporting [24].

To help identify other ecosystem gaps by continuously evaluating attack configurations beyond those considered in our experiments, we are collaborating with the Anti-Phishing Working Group (APWG) to integrate PhishTime as a permanent ecosystem service. Our contributions are thus as follows:

- A framework for the continuous long-term empirical measurement of the anti-phishing ecosystem.
- Deployment of the framework for a longitudinal evaluation of the performance of browser blacklists, with a focus on evasive phishing.
- Identification, disclosure, and remediation of several ecosystem vulnerabilities exploitable by phishers.

[1] Even such a seemingly short delay can cause up to 20% more victims [46].

2 Background

Phishing is a type of social engineering attack [32] through which attackers (known as *phishers*) seek to trick victims into disclosing sensitive information [15]. This stolen information allows phishers to compromise user accounts and identities, which is a significant threat both to the victims and the security of online services [9, 19]. Within the current ecosystem, there exist two main categories of phishing: *spearphishing*, which entails a concentrated effort to trick specific high-value groups or individuals [27], and *large-scale phishing*, which targets a wide range of possible victims and allows phishers to profit through volume [52]. We primarily focus on the latter in this work.

2.1 Phishing Attacks

In a typical phishing attack, phishers first configure and deploy a deceptive phishing website to mimic the appearance of a legitimate website (e.g., of a bank or e-mail provider) that is likely to appear familiar to potential victims. Phishers then start distributing messages to their victims (e.g., via e-mail or SMS spam campaigns) to *lure* them to the phishing website [10, 28]. Such messages will often contain a call to action that suggests a degree of urgency (e.g., correcting a billing error or securing an account) [61]. Victims who are successfully lured will then visit the phishing website and follow its prompts, which may ask for account credentials, financial information, or biographical data. Finally, the data harvested by the phishing website is exfiltrated back to the phishers and can then be used to commit fraud [57].

Phishing attacks have a low barrier to entry and are easy to scale due to the existence of myriad illicit services in underground communities. To deploy phishing websites, many attackers purchase or obtain *phishing kits*, which are all-in-one packages with all the necessary software to create a phishing website [6, 13]. Additional services allow phishers to orchestrate attacks with minimal effort [54, 55, 58].

Although phishing kits vary in quality, the recent growth in phishing volume—which coincides with a decline in malware and drive-by-downloads—has been accompanied by a general increase in sophistication [2, 18, 62]. For example, advanced kits venture beyond stealing account credentials and may ask their victims to provide detailed financial and personal information [46]. Additionally, such kits incorporate features to evade detection by automated anti-phishing systems [44] and may even attempt to intercept two-factor authentication in real time [60]. The threat that phishing poses to victims, organizations, and Internet infrastructure has given rise to an anti-phishing ecosystem that has matured over time—in response to the evolution of phishing—to provide multiple layers of defense [45].

2.2 Anti-phishing Blacklists

Browser blacklists are a key anti-phishing defense that protects users transparently and is enabled by default in most major web browsers across both desktop and mobile

devices [44]. Thus, blacklists are capable of protecting users on the same scale at which phishing occurs.

When a user attempts to visit a phishing website whose URL is known to the browser's blacklist, the browser will display a prominent warning in place of the phishing content [52]. Moreover, blacklists can be integrated with e-mail spam filters to outright prevent users from being exposed to e-mails with the same malicious URL. Blacklists are supported by extensive backend infrastructure that collects suspected phishing URLs and verifies malicious content prior to adding them to the blacklist (to avoid false positives). Some blacklists are also supplemented by in-browser heuristic classifiers [35].

Evasion Techniques. A notable weakness of blacklists is that they are inherently *reactive*. Phishers capitalize on the time gap between a phishing website's deployment and its subsequent blacklisting, and may increase their return-on-investment (ROI) by prolonging this gap [26, 41]. Because blacklist detection relies on content verification, blacklists are vulnerable to evasion techniques which, when successful, may *delay* or *entirely prevent* blacklisting [44]. In Section 6, we describe our approach to testing evasion techniques commonly used in the wild.

Cloaking is an evasion technique that seeks to hide phishing content from blacklist infrastructure (i.e., web crawlers) while keeping it visible to human victims [30]. When a phishing website with cloaking suspects that a request is from a crawler, it will replace the phishing content with a benign-looking page or an error message. Cloaking has become standard in phishing kits, and it is commonly implemented on both the server side and client side by applying filters based on HTTP request attributes and device characteristics [45].

Redirection links make it more difficult for anti-phishing systems (e.g., e-mail spam filters or blacklists) to correlate a link in a lure with a known phishing URL [10]. Because blacklists block phishing websites based on their URLs, phishers typically distribute lures with different URLs that then redirect [20] to the final phishing URL. The HTTP redirection chain itself may implement cloaking to further evade detection, and a many-to-one mapping may exist between redirection links and phishing websites to dilute each link's perceived maliciousness [65]. Phishers commonly abuse URL shortening services to create redirection links [10].

Compromised infrastructure is regularly used by phishers to host phishing kits [1, 31]. Such infrastructure—which otherwise contains legitimate websites—poses a particular challenge to blacklists, as the blacklists must ensure that the legitimate content is not blocked alongside the phishing content (e.g., it might only differ in the *path* of a URL on the same domain [3]). Some phishing kits exploit this phenomenon by generating many sub-folders under one domain, all of which must then be individually blacklisted [46].

Reporting Protocols. Just as individual users rely on browser blacklists to stay safe from phishing, the organizations impersonated by phishing websites rely on blacklists to protect their customers. These organizations typically obtain phishing reports from their customers or internal systems, and then forward the identified URLs to blacklists, either directly or through the help of third-party vendors [48].

Blacklists predominantly accept reports of phishing websites in the form of a bare URL [22, 23, 42, 53]. However, such reports can prove ineffective if the website successfully uses evasion, as the blacklist may mistake the website as benign and thus fail to act appropriately on the report. Reporting protocols that facilitate the submission of additional evidence (e.g., screenshots or page source) are currently available on a limited scale [11]; we test one such protocol in Section 8.6.

3 Blacklist Evaluation Metrics

In this section, we explain the metrics that we use to evaluate blacklists and describe the specific blacklists that we consider throughout the rest of this paper.

3.1 Blacklist Performance

Discovery refers to a blacklist's ability to identify new URLs in the wild that are suspected of hosting phishing content. A blacklist with ideal discovery would know of every URL within the population of live phishing URLs. Discovery can result from direct phishing reports or other ecosystem sources, such as monitoring of e-mail spam, web traffic, website content, or server configuration [5, 17, 35, 43, 46].

Detection refers to a blacklist's ability to correctly classify the *discovered* URLs, such that URLs with phishing content are added to the blacklist. A blacklist with ideal detection would not only flag every true-positive phishing URL, but it would do so promptly at the time of discovery to minimize the potential damage caused by each attack. Thus, we can split detection into two sub-metrics: For any set of phishing URLs discovered by a blacklist, *coverage* is the proportion of these URLs that are blacklisted at any point while they host phishing content. *Speed* is the time delay between discovery and blacklisting, which assesses how quickly blacklists respond. It is thus desirable for blacklists to deliver high coverage and high speed[2].

3.2 Selection of Blacklists

Several different service providers maintain anti-phishing blacklists that are natively included in modern web browsers. *Google Safe Browsing* (GSB) protects Chrome, Safari, Firefox, and Chromium [25]; by global browser market share as of December 2019, GSB is the most impactful blacklist as it protects approximately 80.30% of desktop users and 92.22% of mobile users [8]. *Microsoft SmartScreen* protects Internet Explorer (IE) and Edge [38] and accounts for approximately 12.96% of desktop users. *Opera's fraud and malware protection* leverages undisclosed third-party providers [47, 50] to protect the Opera browser, which has a market share of approximately 1.50% on desktops and 1.27% on mobile. We focus

[2]Perfect detection is nontrivial in part because blacklists must maintain a very low false-positive rate to avoid disrupting legitimate websites [64].

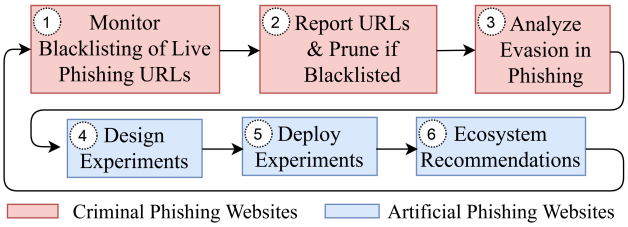


Figure 1: High-level overview of the PhishTime framework.

our evaluation on these top three global blacklists and pay particular attention to GSB due to its large potential impact.

There exist other browser blacklists with a lower global market share but with prominence in specific countries, such as *Tencent Security* and *Yandex Safe Browsing* [8, 56, 67]. In our experiments, we do not consider these blacklists or other anti-phishing systems that are not enabled by default in browsers, such as third-party browser plugins or antivirus software [52]. However, our methodology and framework could be applied to evaluate these alternatives.

4 PhishTime Overview

An effective way to empirically evaluate the performance of anti-phishing blacklists is to deploy a large *batch* of specially-configured test phishing websites, report the websites directly to blacklists, and then monitor each website to see if and when it is blacklisted [44, 48]. For our longitudinal evaluation of blacklist performance, we make a series of such deployments, at regular intervals, over an extended period of time. Within each deployment, we configure multiple distinct batches of websites to support different experiments.

The goal of our experiments is to provide insight into potential gaps within the ecosystem, which could, in turn, lead to actionable security recommendations. We therefore seek to closely replicate the phishing website configurations (i.e., evasion techniques) used by attackers. To identify such configurations and guide our experimental design, we developed the *PhishTime* framework.

We obtained permission from PayPal, Inc. to use PayPal-branded phishing websites throughout our experiments[3]. Therefore, in our PhishTime ecosystem analysis, we also focus on PayPal phishing websites in the wild. Although we were unable to collaborate with other companies for this research, our methodology is generic and could be used for any brand(s).

The PhishTime framework is our systematic, semi-automated approach for identifying evasive (i.e., unmitigated) phishing websites in the wild. We use the framework to characterize both *typical* and *emerging* evasion techniques used by real phishing websites. Understanding the ecosystem's response to typical phishing enables identification of gaps currently being exploited by attackers, whereas analysis of less prominent emerging evasion techniques allows us to take a proactive approach to mitigate the expansion of sophisticated developments in phishing.

The system workflow is described in Figure 1, and proceeds as follows. PhishTime begins by collecting a number of real, live phishing websites (i.e., operated by criminals) for analysis, with Section 5 covering the following steps:

Monitor Blacklisting of Live Phishing Websites. First, we build a sample of live phishing URLs (①) and continuously monitor their status on blacklists of interest. In our deployment, in real time, we collected PayPal phishing URLs from the APWG eCrime Exchange [2] and URLs from phishing e-mails reported directly to PayPal. Using multiple data sources helps increase the diversity of the sample: we found many URLs unique to each respective source, likely due to differences in their data collection and detection approaches.

Report URLs and Prune if Blacklisted. If any URL is not initially blacklisted, we report it (②) directly to the blacklists, and to other key anti-phishing entities, in an effort to get it blacklisted (using the approach and infrastructure described in Section 7.2). We subsequently prune URLs blacklisted within a reasonably short period thereafter and retain those that are not. Recent work has shown that once detected by a blacklist's backend, the majority of phishing URLs show blacklist warnings within two hours [46]. We, therefore, chose a blacklisting cutoff of two hours to eliminate URLs that blacklists could successfully *detect*, but likely originally failed to *discover*.

Analyze (Evasive) Phishing Websites. We then manually inspect the remaining URLs (③) to understand why they have been evading blacklist detection. We analyze the evasion techniques used as well as the behavior (i.e., general appearance and user interface) of the website. We performed this step by first visiting each URL, and then testing different variations of request parameters until we successfully retrieved the content. We can thus infer the server-side evasion techniques used by each phishing website. We also analyze each website visually, and inspect client-side source code, to not only uncover any additional evasion logic, but to compare the websites to samples of known phishing kits available to us to determine which are the most common. Simultaneously, we identify and exclude false positive or offline websites.

The rest of PhishTime's operation leverages the insights extracted from criminals' phishing websites and, through the automated deployment of our own artificial phishing websites that mimic them, achieves continual monitoring of blacklist performance.

Design (Evasion-inspired) Experiments. After analyzing a representative sample of URLs, we abstract the key trends that we observed and design experiments to replicate them in a controlled setting (④, Section 6).

[3] In the current ecosystem, PayPal is among the brands most commonly targeted by phishers [59].

Deploy PhishTime Experiments. Finally, we deploy these experiments (⑤, Section 7) to evaluate blacklist performance, over time, in the face of diverse evasion.

Ecosystem Recommendations. We use our experimental results to make security recommendations for specific blacklists or the ecosystem (⑥, Sections 8-9). Any resulting ecosystem changes can then influence the design of experiments in successive framework deployments.

5 PhishTime Analysis

We used the PhishTime framework in January 2019 to identify phishing websites in the wild capable of successfully evading blacklisting for extended periods of time. We then characterized *typical* evasion techniques used by these websites, and we designed experiments which entailed deploying a series of PhishTime-crafted phishing websites to empirically measure the response of blacklists to these techniques, in a controlled manner. Later, in August 2019, we used the framework to identify less common (but more sophisticated) *emerging* evasion techniques, and we designed additional experiments to test these techniques. We show a timeline of our ecosystem analysis using PhishTime, and the subsequent experiment deployments, in Figure 2.

5.1 Typical Evasion Techniques

In total, we analyzed 4,393 distinct phishing URLs in the wild and found that 183 failed to be promptly blacklisted. Although this may seem like a relatively small number, prior work has shown that the majority of real-world damage from phishing occurs from a small fraction of known phishing URLs [46]. Moreover, the total URL count for the ecosystem would be considerably higher, as we focused only on a single brand.

Of these 183 websites, 96 were never blacklisted anywhere before going offline (the average observed lifespan was 17 hours, 12 minutes), 87 were ultimately blacklisted in at least one desktop browser (with an average observed speed of 7 hours, 4 minutes) and 23 were ultimately blacklisted in at least one mobile browser (with an average observed speed of 12 hours, 2 minutes). We also observed 10 websites which remained live, without blacklisting, for over one week. Note that due to the inherent delay between an attacker's deployment of a phishing URL and its appearance in a report or feed, the aforementioned timings represent lower bounds [34].

By analyzing URLs in the e-mail lures reported to PayPal, we found that 177 of these websites had *lure* URLs which redirected to a different final landing URL with the phishing content. We observed redirection URLs both through third-party redirection services and attacker-controlled domains. In the latter case, we commonly observed JavaScript-based redirection alongside traditional HTTP redirection [20]. We also observed that at least 146 of these websites used some form of server-side cloaking [45]: we were unable to retrieve their content using a cloud-based web crawler but succeeded when using a mobile IP address or anonymous VPN service.

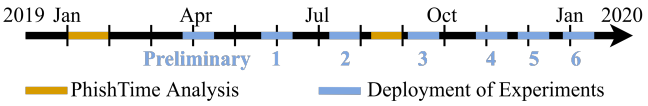


Figure 2: Timeline of framework & experiment deployments.

At least 42 websites had a different URL path or a subdomain of a domain that appeared in another phishing website, which reflects phishers' tendency to re-use infrastructure.

5.2 Emerging Evasion Techniques

We found that eight of the 96 phishing websites which were never blacklisted implemented clever mechanisms to evade detection: five presented visitors with a CAPTCHA challenge prior to displaying the phishing content, two required the user to click on a button in a popup window prior to redirecting to the phishing page, and one would not render content prior to detecting mouse movement. We refer to these evasion techniques as *behavior-based* because they require a specific user behavior to display phishing content.

6 Experimental Design

We now transition from merely observing the ecosystem to actively measuring it: to methodically test the phishing website configurations we observed, we replicate them across a large sample of our own new artificial phishing websites. We deploy these websites, report the respective URLs to several anti-phishing entities, and monitor the speed and coverage of blacklists as they respond to our reports. We conducted our experiments ethically, to avoid harming any real users or anti-phishing systems, as discussed in Section 9.2.

In total, we made one preliminary deployment in March 2019 and six main deployments of experiments at regular intervals between May 2019 and January 2020. The purpose of the preliminary deployment—which mirrored the configuration of the first main deployment—was to verify the soundness of our experimental design and the technical correctness of our framework. We summarize our deployments in Table 1.

Across the six main deployments, we launched 2,862 phishing websites as part of seven different experiments. We registered a total of 2,646 new *.com* domain names for these websites. Because some of our experiments involved redirection links, an additional 1,296 such links bring our overall URL count to 4,158. As our experiments seek to make several distinct measurements over time, each deployment includes multiple different experiments.

Each experiment consists of one or more distinct *batches* of phishing websites: groups that share a single configuration corresponding to the respective *experiment*. We chose our batch size, 54, by estimating the required number of domains (i.e., which we would then purchase) for a sample size that could support statistically significant inferences in one-way ANOVA among sets of batches: To obtain a power of 0.95 at a p-value of 0.05, we initially assumed a *medium* effect size of 0.25 [12]. Using the baseline GSB blacklist speed observed

Experiment		Deployment						Per Deployment			Total		
		1 May	*2 Jul.*	*3 Sep.*	*4 Oct.*	*5 Nov.*	*6 Dec.*	*Batches*	*Websites*	*URLs*	*Batches*	*Websites*	*Domains Registered*
A	*Baseline*	✓	✓	✓	✓	✓	✓	1	54	54	6	324	324
B	*Basic Evasion*	✓	✓	✓	✓	✓	✓	1	54	54	6	324	324
C	*Typical Evasion (Redirection)*	✓	✓	✓	✓			3	162	324*	12	648	1,080
D	*Domain Reuse*	✓	✓	✓	✓			3	162	324*	12	648	0
E	*Discovery*	✓	✓	✓	✓			2	108	108	8	432	432
F	*Emerging Evasion*					✓		7	378	378	7	378	378
G	*Evidence-based Reporting*						✓	2	108	108	2	108	108

Table 1: Experiments conducted during each of our main deployments (*half are redirection links).

in the preliminary deployment, we calculated a higher effect size of 0.36, which suggests an adequate sample size selection.

6.1 Measuring Blacklist Speed & Coverage

The experiments in this section focus primarily on measuring the *detection* performance (i.e., speed and coverage) of blacklists. As we believe that it is generally infeasible for attackers to avoid *discovery* when conducting traditional phishing attacks (e.g., at scale through e-mail spam), our reporting methodology seeks to ensure that all URLs we deploy as part of these experiments are promptly discovered by the blacklists we test. We do so by simultaneously reporting the URLs to multiple blacklists and other anti-phishing entities, which we elaborate on in Section 7.2.

Experiment A: Baseline. For our simplest experiment, we launch a single batch of basic phishing websites, with no evasion technique, once in each deployment. These, and all other websites we deploy, used HTTPS to match current ecosystem trends [2]. This experiment serves two key purposes: to establish a baseline for the best-case speed and coverage provided by blacklists (for comparison to other experiments), and to measure if these metrics remain consistent over time.

Experiment B: Basic Evasion. In this experiment, we test two straightforward cloaking techniques inspired by our observations in Section 5.1: websites that only allow traffic from browsers with a mobile *user agent* [20, 30], and websites that render content using JavaScript. We alternate these two cloaking techniques between deployments.

This experiment allows us to evaluate blacklists' response to slightly more sophisticated phishing by comparing against the baseline response in Experiment A. It also establishes a point of comparison for even more sophisticated phishing in later experiments. A secondary objective of this experiment is to assess blacklist coverage (on mobile devices) of phishing websites aimed specifically at mobile users. Mobile devices have historically been prone to phishing [63], and recent work has revealed gaps in blacklisting on mobile devices [44].

Experiment C: Typical Evasion (Redirection). Each deployment in this experiment has three batches of websites that focus on evaluating the evasiveness of redirection. In a one-to-one mapping, we pair each phishing website with a *different* URL that redirects to it with an HTTP 302 status code [20]. For this experiment, we *only* report the redirection URLs (i.e., the URLs that could serve as lures in a phishing e-mail). We configured each phishing website with the same evasion technique as Experiment B in the respective deployment.

In the first of the three batches, we used a popular link shortening service, *bit.ly*, to generate the redirection links. Such services are commonly used by attackers to scalably generate unique lures. In the second of the three batches, we used our own *.com* domains (each different from the website's domain) for the redirection links. In the third batch, we similarly used *.com* domains for the redirection links, but additionally configured them with server-side IP and hostname cloaking [45]. The latter batch thus most closely mirrors the typical configuration of the phishing websites that we observed in Section 5.1; we based the cloaking technique on the *.htaccess* file (which blocks known crawlers) found in a phishing kit that we commonly observed in the wild during the PhishTime analysis (③).

Because we only change one variable between the three batches, we can compare the blacklisting of phishing websites that combine redirection with cloaking on *both* the lure and the phishing website with the blacklisting of websites with lesser degrees of evasion. We can also evaluate the feasibility for attackers to use, and the ecosystem's mitigation of, third-party redirection services.

Experiment D: Domain Reuse. After the completion of each Experiment C deployment, we generate identical batches of websites on the same domains as in Experiment C, but with different *URL paths* [3]. We then redeploy these websites as part of a new experiment, which seeks to measure how blacklist speed and coverage change when phishers re-use domains and infrastructure to carry out successive attacks (a strategy phishers use to increase their ROI).

Experiment F: Emerging Evasion. These websites mirror the sophisticated, emerging evasion techniques we observed in Section 5.2. Three batches implement evasion using JavaScript code that we found in the wild for CAPTCHA, popup, and mouse movement cloaking, respectively. Three additional batches have the same configuration but with added *.htaccess* server-side cloaking, as in Experiment C. One final batch had only *.htaccess* cloaking, as a control group.

6.2 Other Measurements

Our remaining experiments follow a different reporting methodology than those in the previous section.

Experiment E: Discovery. In this experiment, we launch

two batches of websites, per deployment, that mirror the (basic) configuration of Experiments A and B. However, we only report each batch to a single anti-phishing entity (PayPal or the APWG), alternating between deployments. Thus, by comparing against Experiments A and B, we can evaluate how well our primary reporting methodology ensures prompt *discovery* by blacklists. We can also directly test the performance of specific anti-phishing entities: we chose PayPal's own anti-phishing system because our websites used the PayPal brand, and we chose the APWG because it had been shown to reliably share phishing URLs with other entities [2, 44].

Experiment G: Evidence-based Reporting. When we initially designed our experiments, Google Safe Browsing only allowed the submission of bare URLs when reporting phishing (whether manually or programmatically). However, in July 2019, with the release of the Chrome Suspicious Site Reporter (CSSR) [11] plugin, manual reports could be enhanced with additional evidence: a screenshot, source code, and the redirection chain, IP address, and user agent for the request. To evaluate if this enhanced reporting approach could help improve blacklists' detection of evasive URLs, we designed this additional experiment to compare the coverage of GSB when reporting with the old and the new method.

We configured the two batches of phishing websites in this experiment with cloaking that limits traffic to US IP geolocations: a strategy that was recently capable of evading GSB [44]. We reported one batch via CSSR [11] and the other batch via the traditional GSB URL submission form [22]. Because CSSR only supports manual submissions, we compared it to another manual submission channel.

7 Implementation of Experiments

We adapted a previously-proposed testbed (*PhishFarm* [44]) to deploy the phishing websites needed for each of our experiments. The testbed enables the automated configuration, deployment, and monitoring of *innocuous* but real-looking phishing websites to empirically measure browser-based defenses such as blacklisting. To accurately emulate current phishing trends and ecosystem defenses, we enhanced the testbed to support automation of HTTPS website hosting, lures with redirection, and API-based reporting.

7.1 Overview

In Figure 3, we provide an overview of the steps we took to deploy each experiment. First, we prepare the hosting infrastructure (Ⓐ). We used the aforementioned testbed to host our phishing websites on 45 cloud-based Apache web servers, each with a unique US IP. At the time of each deployment, we configure DNS records to point the required domains to these web servers, and we install Let's Encrypt SSL certificates [33] for each domain. Next, we configure the phishing website content and behavior (i.e., evasion techniques) for each URL, and we test this configuration to verify the correct operation of the framework (Ⓑ). We then activate the websites and

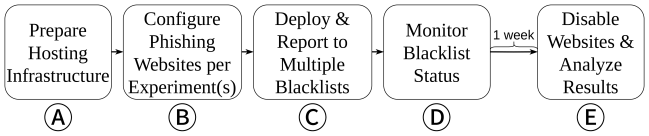
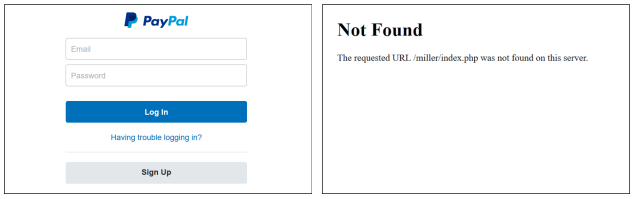


Figure 3: Steps in each deployment of experiments.



(a) Successful request (b) Request denied by cloaking

Figure 4: Appearance of our phishing websites.

immediately report their URLs to the anti-phishing entities specified by the experimental design (Ⓒ). Over the course of the next seven days, we monitor the blacklist status of our URLs and we collect web traffic metadata (Ⓓ). Finally, we deactivate the websites and analyze the collected data (Ⓔ). Each of these steps is fully automated by the testbed.

All of our phishing websites matched the look and feel of the *PayPal.com* login page as it appeared in January 2019. Whenever a crawler request was denied by the cloaking technique on a particular website, it would encounter a generic *404* error message [20], as shown in Figure 4.

7.2 Reporting to Blacklists

To maintain consistency across our large number of experiment deployments, we fully automated our reporting methodology. Our reporting approach is representative of the actions that an organization targeted by phishers might take to mitigate known phishing websites [44].

To report each of our phishing websites, we submit its URL directly to Google Safe Browsing via the Phishing Protection Submission API [24][4] and to the APWG via the eCrime Exchange API [2]. Direct API reporting is not available for Opera and Microsoft SmartScreen. However, prior work has shown that the APWG and other major anti-phishing entities share data with these blacklists [44, 50]. Therefore, we report to these additional entities via e-mail. Using a PayPal-branded phishing e-mail template found in the wild, we generate a fake phishing e-mail with the URL of the website. We then forward this e-mail as an attachment to anti-phishing entities that accept reports from the public: PhishTank [49], Netcraft [42], PayPal [51], and US CERT [21]. This reporting methodology seeks to guarantee all blacklists' *discovery* of our phishing websites (thus, it does not apply to Experiments E and G, as previously discussed in Section 6.2).

[4] At the time of our deployments, the Phishing Protection Submission API was in a beta stage and not available to the public. Google provided us with access to the API for this research.

7.3 Blacklist Monitoring

We used a total of 40 virtual machines (VMs) to empirically monitor blacklisting of each website at least once every 10 minutes across six desktop browsers: Chrome, Safari, Firefox, IE, Edge, and Opera. In addition, to determine the speed of blacklisting on mobile, we monitored Google Safe Browsing programmatically using the Update API [23]. Using a single physical Android phone (connected to the Internet over Wi-Fi), we also empirically compared the coverage of mobile Chrome, Firefox, and Opera to their desktop counterparts.

7.4 Experimental Controls

To ensure the validity of our experimental data, we meticulously controlled the configuration and deployment of our experiments to minimize the effect of confounding factors on the observed speed of blacklisting: any factors other than the evasion technique of each website (Experiments A-F) or the reporting channel (Experiment G).

Website Metadata. Beyond classifying phishing websites based on their content, anti-phishing systems (including blacklists) consider metadata such as deceptive URL keywords, domain age, and URL and IP reputation [64]. Each of our domains and URL paths consisted of combinations of random English words to limit detection via URL or DNS attributes [5, 68]. To ensure that no historical maliciousness was associated with our phishing URLs, we registered a new domain name for each URL reported (except Experiment D, which deliberately measured domain re-use). We also registered our domains six months before each experiment, leveraged a major registrar (GoDaddy), and used the *.com* TLD (found in the majority of current phishing URLs) to minimize detectability through these attributes [2].

Network Traffic. To prevent network signals from our monitoring infrastructure from potentially skewing blacklisting, our websites showed benign content to requests from this infrastructure. We also disabled client-side anti-phishing features in the browsers used for monitoring. Similarly, queries to the Update API did not leak the URLs being checked.

Consistent Reporting. Some anti-phishing systems filter reports to mitigate the risk of being flooded by fictitious URLs from attackers. Our direct reports through Google's non-public API inherently avoid such filtering. Also, each of our e-mail reports originated from a different e-mail address, and information such as the (fictitious) victim name or transaction amount was randomized between reports. We initiated each deployment at approximately the same time of day. We then sent the reports for any given experiment in a single pass to minimize variations in reporting time, and we throttled the reports to avoid an excessive reporting rate.

Experimental Variables. Within each experiment, we varied the configuration of different batches in at most one way to be able to perform a comparative analysis on a single variable. The same concept also applies *between* the majority of our experiments, which can thus collectively paint a multi-dimensional view of the response of anti-phishing blacklists.

Experiment Duration. Anti-phishing blacklists typically respond within a matter of hours; however, in certain cases (e.g., due to cloaking), blacklisting may be delayed by several days as additional (possibly manual) checks are made by various entities [44]. This observation, combined with occasional long-lasting phishing websites during the PhishTime analysis, motivated our conservative choice of a one-week lifespan for each phishing website in our experiments.

We discuss possible trade-offs in our controls in Section 9.3. Nevertheless, in the following section, we show that our experiments generally led to a consistent response by the ecosystem and ultimately yielded actionable findings.

8 Experimental Results

After the completion of all our experiment deployments, we had collected extensive data for each of the 4,158 URLs that we launched and monitored: timestamps of blacklisting (in six desktop browsers, three mobile browsers, and the Google Safe Browsing API), online status, certificate revocation status, and web traffic logs. Our infrastructure operated as expected during each main deployment.

In the analysis that follows, for any given batch of URLs, we define the coverage of a given blacklist as the percentage of all URLs that were blacklisted *at any point* during the seven-day deployment of the batch. For any given URL, we define blacklist speed as the elapsed time between *our* reporting of that URL and its subsequent blacklisting. Within an individual batch, we either provide median speed in tabular form or plot speed as a function of coverage over time.

Simplification of Dimensionality. Our empirical monitoring of desktop browsers revealed that Chrome and Safari consistently delivered the same blacklist speed and coverage, whereas Firefox was an average of 10 minutes slower (likely stemming from different caching of the GSB Update API [24]) but still had the same coverage. Similarly, in comparing IE and Edge across all deployments, we found that the former was 12 minutes slower on average, also with the same coverage. Thus, to simplify and clarify our analysis, we exclude the desktop versions of Safari, Firefox, and IE from our evaluation.

On mobile devices, we found the blacklist speed and coverage of Firefox to be identical to its *desktop* counterpart. Offline verification of the GSB API data also showed that mobile Safari was consistent with mobile Chrome. We therefore do not duplicate the respective metrics in the tables in this section. However, neither mobile Chrome nor mobile Opera showed consistency with their desktop versions. Note that due to limited mobile hardware, we could not accurately measure the speed of mobile Opera across all experiments, so we exclude this data.

Data Aggregation. We aggregate our blacklist measurements based on the objectives of each experiment, as defined in Section 6. For longitudinal comparisons, we group blacklist performance by deployment; to evaluate evasion, we aggregate multiple deployments by experiment or batch.

		Desktop						Mobile				Avg. Traffic	
		GSB		*SmartScreen*		*Opera*		*GSB: Chrome/Safari*		*GSB: Firefox*	*Opera*	*All Requests*	*Successful Requests*
Deployment		Coverage	Median Speed	Coverage	Median Speed	Coverage	Median Speed	Coverage	Median Speed	Coverage	Coverage		
1	May 2019	100.0%	00:44 (hh:mm)	100.0%	02:02	98.1%	00:37	100.0%	09:19	100.0%	0.0%	1677	1151
2	Jul. 2019	100.0%	00:51	100.0%	02:38	70.4%	00:32	55.6%	35:28	100.0%	0.0%	7003	1491
3	Sep. 2019	64.8%	00:50	61.1%	04:44	22.2%	01:52	13.0%	159:22	64.8%	14.8%	286	211
4	Oct. 2019	98.1%	01:00	100.0%	02:19	64.8%	00:55	50.0%	03:05	98.1%	14.8%	3756	2020
5	Nov. 2019	100.0%	01:26	100.0%	02:27	59.3%	00:38	13.0%	39:11	100.0%	0.0%	1566	682
6	Dec. 2019	100.0%	00:46	100.0%	02:34	48.1%	00:28	70.4%	00:28	100.0%	9.3%	3255	1554

Table 2: Blacklist performance vs. unevasive phishing (Experiment A: raw data for each deployment).

		Desktop						Mobile				Avg. Traffic	
		GSB		*SmartScreen*		*Opera*		*GSB: Chrome/Safari*		*GSB: Firefox*	*Opera*	*All Requests*	*Successful Requests*
Deployment		Coverage	Median Speed	Coverage	Median Speed	Coverage	Median Speed	Coverage	Median Speed	Coverage	Coverage		
1	May 2019	89.5%	02:29 (hh:mm)	88.0%	10:20	83.4%	03:54	54.9%	07:36	89.5%	0.0%	2038	603
2	Jul. 2019	99.3%	01:46	99.5%	05:42	43.0%	01:41	0%	-	99.3%	31.8%	508	53
3	Sep. 2019	79.9%	02:21	69.5%	08:24	50.1%	02:36	3.2%	34:47	79.9%	29.3%	1073	589
4	Oct. 2019	86.7%	01:32	90.0%	10:19	58.2%	01:51	0%	-	86.7%	35.0%	545	45

Table 3: Blacklist performance vs. evasive phishing (Experiments B, C, D: average of all deployments).

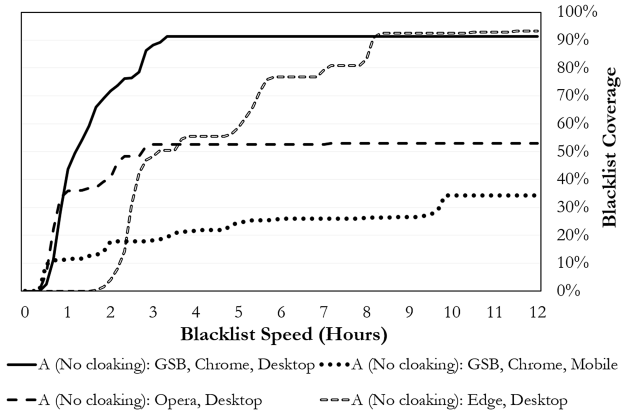


Figure 5: Aggregate speed and coverage of all blacklists against uncloaked websites (Experiment A, Deployments 1-6).

8.1 Discovery

Of the 4,158 URLs that we deployed, 4,068 received traffic from at least one crawler. The 94 URLs which were never visited were all part of Deployment 3: 81 URLs were part of Experiment E (reported to a single entity) and 13 were post-redirection landing pages within Experiment C.

3,514 of our URLs were blacklisted in at least one browser. Of the 644 URLs never blacklisted, 299 were part of Experiment F (in which sophisticated cloaking successfully evaded detection), 131 were part of Experiments E or G (which were not designed to guarantee discovery), and 214 were part of Experiments B, C, and D (with cloaking and redirection).

Given that the aforementioned lack of traffic can be attributed to the ecosystem issues we identified during Deployment 3 (discussed in Section 8.2), and the fact that all websites from Experiment A were blacklisted in at least one browser, we believe that our reporting methodology was successful in ensuring prompt discovery by the ecosystem.

8.2 Overall Blacklist Performance

In Table 2, we show the blacklist speed and coverage results from each of the six deployments of Experiment A, as well as the average number of crawler requests to *each* individual website. Because this experiment consisted solely of unsophisticated phishing websites without any form of evasion, it allows us to establish a baseline for best-case blacklist performance which we can compare against other experiments.

Desktop Blacklists. With an average coverage of 92.9% and an average speed of 56 minutes (based on the medians across our six deployments), overall, GSB proved to be the best-performing blacklist we tested. SmartScreen showed a slightly higher coverage of 93.2%, but had a slower speed of 3 hours and 47 minutes. Opera's coverage was the lowest, at 60.5%, though its 55-minute speed managed to inch ahead of GSB.

Mobile Blacklists. The mobile version of Firefox mirrored the 92.9% coverage of GSB on desktop and had the highest coverage of the mobile blacklists we tested. Mobile Chrome and mobile Safari delivered a much lower coverage of 57.8%, whereas Opera's coverage was minimal at 3.7%.

Although aggregate speed and coverage metrics provide an assessment of overall blacklist performance, they fail to illustrate specific differences in behavior between blacklists. In Figure 5, we plot the growth of each blacklist's coverage over the course of the first 12 hours of our deployments (the same data over the full one-week deployment is in Figure 6). We observe that GSB and Opera both start blacklisting as early as 20 minutes after receiving our phishing reports. SmartScreen's earliest response occurred about one hour after GSB and Opera, and grew over a seven-hour period thereafter. On desktop platforms, GSB coverage grows quickly and stabilizes after approximately three hours; on mobile devices, coverage grows more slowly over a longer period and is a subset of the desktop blacklisting. We did not observe any patterns in our website configurations that consistently led to mobile blacklisting, nor did such websites receive more crawler traffic.

Long-term Blacklist Consistency. High blacklist speed and coverage are necessary to effectively protect users from phishing websites, but, given the pressure upon the ecosystem by phishers [2, 25], it is equally important that blacklist performance remains consistent in the long term. By comparing the measurements between successive deployments (in Table 2 and 3 for non-evasive and evasive phishing websites, respectively), we can evaluate this consistency.

Per the data for Experiment A, we observe that both GSB and SmartScreen delivered 100% coverage and similar speed in five

		Desktop						Mobile				Avg. Traffic	
		GSB		*SmartScreen*		*Opera*		*GSB: Chrome/Safari*		*GSB: Firefox*	*Opera*	*All Requests*	*Successful Requests*
Experiment	**Batch**	Coverage	Median Speed	Coverage	Median Speed	Coverage	Median Speed	Coverage	Median Speed	Coverage	Coverage		
Experiment A (Baseline)		92.9%	00:57 (hh:mm)	93.2%	02:48	60.5%	00:55	57.8%	17:30	92.9%	3.7%	3452	1366
Experiment B (Basic Evasion)	JavaScript Cloaking	88.3%	01:03	100.0%	03:30	49.4%	00:56	0.0%	-	88.3%	0.0%	455	115
	Mobile Cloaking	100.0%	00:55	100.0%	02:39	44.0%	00:38	0.0%	-	100.0%	0.0%	936	207
Experiment C (Typical Evasion - Redirection)	bit.ly Redirection - Lure	86.1%	01:25	91.4%	03:02	46.3%	01:40	0.0%	-	86.1%	0.0%	2313	2313
	bit.ly Redirection - Landing	86.1%	02:58	88.0%	12:45	59.3%	02:46	25.9%	43:51	86.1%	25.0%	593	392
	.com Redirection - Lure	83.3%	01:44	99.4%	03:09	50.6%	01:57	0.0%	-	83.3%	41.4%	440	81
	.com Redirection - Landing	88.9%	02:48	87.0%	09:35	59.7%	02:55	24.1%	11:46	88.9%	30.6%	740	454
	.com Redirection w/ .htaccess	80.2%	01:36	77.2%	08:51	37.7%	01:31	0.0%	-	80.2%	25.3%	275	28
	.com Redirection w/ .htaccess - Landing	84.3%	02:43	86.6%	10:01	51.9%	02:33	9.3%	11:19	84.3%	32.9%	370	63
Experiment D (Domain re-use)	bit.ly Redirection - Lure	96.3%	01:09	94.4%	06:51	58.0%	00:41	0.0%	-	96.3%	0.0%	5143	5143
	bit.ly Redirection - Landing	97.2%	02:03	72.7%	11:56	58.0%	02:21	4.3%	00:01	97.2%	52.5%	876	497
	.com Redirection - Lure	95.7%	01:10	99.4%	06:59	73.5%	27:24	0.0%	-	95.7%	54.9%	1582	984
	.com Redirection - Landing	98.1%	02:10	71.3%	11:48	66.7%	01:50	3.7%	46:28	98.1%	50.0%	1061	534
	.com Redirection w/ .htaccess - Lure	93.8%	01:13	77.2%	10:07	37.7%	00:57	0.0%	-	93.8%	37.0%	1051	583
	.com Redirection w/ .htaccess - Landing	95.4%	02:17	67.3%	12:19	45.7%	01:53	0.0%	-	95.4%	40.7%	332	42
Experiment E (Discovery)	Reported to APWG	98.1%	02:47	100.0%	02:29	41.7%	02:41	51.9%	04:53	98.1%	41.7%	2901	1591
	Reported to PayPal	16.2%	01:06	38.4%	02:43	6.5%	00:49	13.0%	00:35	16.2%	2.8%	450	293
Experiment F (Emerging Evasion)	Mouse Movement Cloaking	0.0%	-	0.0%	-	0.0%	-	0.0%	-	0.0%	0.0%	37	34
	CAPTCHA Cloaking	0.0%	-	42.6%	03:06	0.0%	-	0.0%	-	0.0%	0.0%	47	42
	Notification Cloaking	0.0%	-	0.0%	-	0.0%	-	0.0%	-	0.0%	0.0%	48	41
	.htaccess Cloaking	100.0%	01:37	100.0%	10:47	42.6%	00:40	0.0%	-	100.0%	0.0%	702	86
	Mouse Movement Cloaking w/.htaccess	0.0% coverage										59	20
	CAPTCHA Cloaking w/.htaccess											45	19
	Notification Cloaking w/.htaccess											48	21
Experiment G (Reporting Methods)	Standard URL Report	20.4%	00:38	0.0%	-	0.0%	-	20.4%	00:17	20.4%	0.0%	5	2
	Chrome Suspicious Site Reporter (CSSR)	90.7%	10:13	0.0%	-	0.0%	-	90.7%	10:17	90.7%	0.0%	16	14

Table 4: Blacklist performance aggregated by each batch (average of all deployments).

of the six deployments. Opera remained consistent in terms of speed across five deployments. For GSB and SmartScreen, we applied ANOVA to each respective set of raw baseline desktop blacklist speed observations (as aggregated in Table 2), treating each deployment as a separate group. We found the differences to be statistically significant, with a p-value below 0.01 for both tests. We believe that these variations—even if relatively small—stem from the complexity of the underlying anti-phishing systems, the order in which reports are processed, and data sharing methods across the ecosystem [64]. However, except for GSB in mobile Firefox, blacklists in *mobile* browsers did not prove to be consistent with their desktop counterparts.

Also, notably, coverage dropped dramatically during Deployment 3, both for non-evasive and evasive phishing, as shown in Tables 2 and 3. In analyzing this anomaly, we first ruled out technical issues and confirmed that all of our e-mail and API reports were successfully delivered. We also redeployed Experiment A with prior domains and reproduced the degraded coverage. After analyzing the results of single-entity reporting in Experiment E (summarized in Table 4), we found that the coverage from reports sent directly to PayPal was similarly low: its coverage in GSB was 9.3% in Deployment 3, down from 44.4% in Deployment 1. Upon comparing crawler traffic between these deployments, we found that crawler activity as a result of PayPal reports was absent from the majority of websites in Deployment 3. Although we cannot rule out other ecosystem factors, we believe that this absence was a key cause of the overall coverage drop, and we disclosed it to PayPal (we later received acknowledgment of the issue, which was ultimately resolved).

Baseline coverage recovered in subsequent deployments, except for a single website in Deployment 4 that failed to be blacklisted by GSB. Despite being blacklisted by SmartScreen and crawled by numerous other entities, it was never crawled by GSB: our original GSB report was likely never acted on, and GSB did not appear to discover the URL through the other entities to which we reported. Though an outlier, this suggests that the ecosystem may benefit from more robust data sharing.

Blacklist Persistence. Across all of our deployments, once a URL was blacklisted in a particular blacklist, we did not observe de-blacklisting during the deployment or within one week immediately thereafter. After the conclusion of our final deployment, we served *404* errors from all of our URLs and monitored them for an extended period. We found that the earliest removal from blacklists occurred 29 days after we had originally reported the respective URL.

We suspect that de-blacklisting may depend on factors such as the presence of benign content on a domain, the domain's reputation, or whether the web server is online. Although our experiments were not designed to pinpoint criteria for de-blacklisting, we believe that the premature removal of phishing URLs from blacklists is currently not a significant issue.

8.3 Typical Evasion Techniques

In Table 4, we show blacklist performance for the specific batches within each experiment, aggregated across all deployments. This allows us to compare speed and coverage when blacklists are faced with different evasion techniques.

Websites with mobile user agent cloaking (in Experiment B) had a negligible effect on desktop blacklisting compared to the unevasive Experiment A (if we disregard the skew from Deployment 3): modern blacklists can, in fact, reliably detect phishing websites with certain basic evasion techniques. Interestingly, however, both GSB and Opera had 0% coverage on mobile devices across all deployments of Experiment B, which is very undesirable given that Experiment B websites were configured to be accessible exclusively on mobile devices. In Figure 6, we visualize blacklisting of these websites in each blacklist over the full duration of our deployments.

In Experiment C, we tested the addition of three types of redirection on top of the basic evasion techniques in Experiment B. For brevity, we focus our discussion on blacklisting by GSB on desktop, and we use the average speed and coverage across all deployments of Experiment B (00:59 and 94.2%, respectively), calculated per Table 4, as a point of comparison. On average, redirection via *bit.ly* links slowed

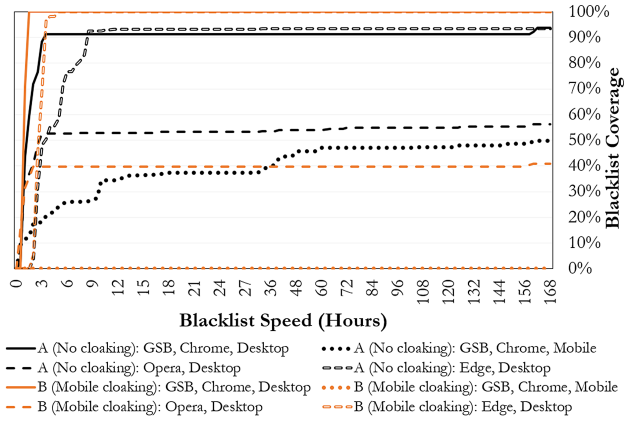


Figure 6: Comparison of all blacklists' aggregate performance for uncloaked websites vs. websites with Mobile cloaking.

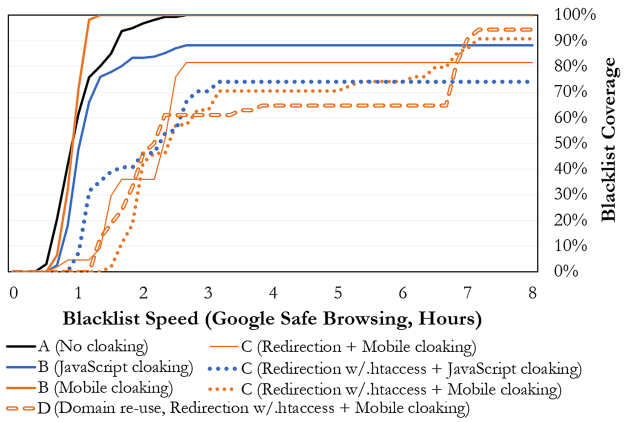


Figure 7: Comparison of aggregate speed and coverage of GSB against the different evasion techniques we tested.

blacklisting speed of the corresponding landing pages to 02:58, and reduced coverage to 86.1%. Redirection via *.com* domain names slowed the speed to 02:48 and reduced coverage to 88.9%. By adding *.htaccess* cloaking on top of redirection, speed only slowed to 02:43, but coverage fell further to 84.3%. As shown in Table 4, the blacklisting speed of the corresponding lures was at least 1 hour faster in each case; however, attackers' ability to easily generate many lures increases the importance of blacklisting the actual landing pages [61].

In Experiment D, we re-deployed phishing websites on the same *.com* domains as in Experiment C, but with different paths, to simulate how attackers might re-use compromised domains in the wild. Although we observed increased speed and coverage compared to Experiment C, the speed remained slower than in experiments without redirection. Only 4.3% of URLs in Experiment D were blacklisted immediately upon deployment, which may represent a gap exploitable by phishers. In Figure 7, we visualize the difference in GSB desktop blacklisting of the cloaking techniques considered in this section. To maintain consistency, we exclude Deployment 3 from the figure. For clarity, we also omit the batches with *bit.ly* links, as they followed the same trend as *.com* redirection links, and were only blacklisted slightly more slowly.

In mobile Chrome and mobile Safari, Experiment C coverage ranged from just 9.3% to 25.9% and was 8 to 40 hours slower than on desktop. Only landing pages, rather than lures, were blacklisted. Interestingly, in Experiment D, coverage dropped to a range of 3.7% to 4.3%, despite the ecosystem's knowledge of our domains from previously blacklisted URLs. We discuss the implications of these inconsistencies in Section 9.

Overall, we observe that delays and gaps exist in the blacklisting of typical phishing websites: these gaps provide a prime opportunity for attackers to successfully target their victims [46], help explain the prevalence of evasion techniques, and should be addressed by the ecosystem.

Disabling of Bit.ly Links. To deter abuse, *bit.ly* disables redirection links that point to known malicious content. During Deployment 1, *bit.ly* disabled 98.1% of the links within Experiment C, and 88.9% of the links within Experiment D, with an average speed of 11:36 (far slower than the tested blacklists). After we switched to a new (non-disabled) API key for subsequent links, no other links were disabled during our research, except for a single URL during Deployment 3. We disclosed these findings to *bit.ly* but received no response.

8.4 Emerging Evasion Techniques

None of the batches of sophisticated cloaking techniques within Experiment F saw any blacklisting, except for one batch with CAPTCHA cloaking, which had 42.6% coverage in SmartScreen only (shown in Table 4). Upon further investigation, we discovered that SmartScreen's detection occurred due to its classification of obfuscation within the CAPTCHA JavaScript code as malware. Because such detection can trivially be bypassed [66], we believe that behavior-based evasion techniques represent a threat to the anti-phishing ecosystem.

A fundamental distinction between the advanced cloaking techniques in Experiment F and the other experiments is that they require *interaction* from a human user to trigger the display of phishing content (i.e., clicking on a button, solving a CAPTCHA challenge, or moving the mouse). Such behaviors might be typical of a human user (and may not even raise suspicion if the user is successfully fooled by an e-mail lure, or if the landing page matches the look-and-feel of the impersonated organization). However, web crawlers would need to be specially developed to emulate such behaviors or otherwise fingerprint such cloaking.

8.5 Single-entity Reporting

In Experiment E, we observed clear differences in blacklist response when comparing reporting to the APWG (only) with reporting to PayPal (only), as shown in Table 4. Even if we exclude the problematic performance of PayPal during Deployment 3 (as discussed in Section 8.2), reporting to the APWG resulted in higher coverage across all blacklists and more crawler traffic to each website. However, the speed of GSB blacklisting after reporting to PayPal was 01:31 faster than

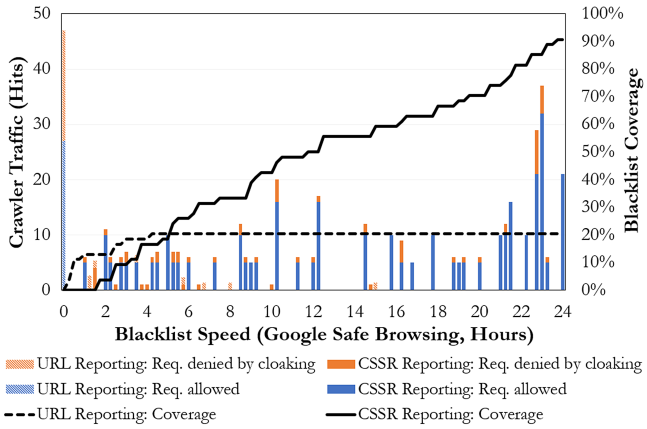


Figure 8: Comparison of traditional URL-only reporting with *evidence-based* reporting in Google Chrome (Experiment G).

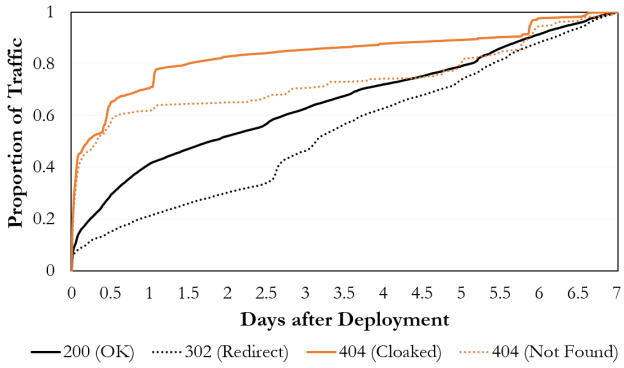


Figure 9: Cumulative Distribution Function of crawler traffic to our phishing websites, across all deployments.

that of the APWG. This suggests that between entities, there exist different implementations for propagating reported URLs to blacklists. Due to each entity's unique strengths, we believe it is important to report phishing to multiple entities.

8.6 Evidence-based Reporting

In Figure 8 and Table 4, we compare the difference in GSB speed and coverage between traditional URL reporting and evidence-based reporting through CSSR [11] from Experiment G (note that we limit the *x-axis* in the figure as coverage did not increase after 24 hours).

We observe that the two reporting approaches each resulted in a distinct crawler response and subsequent blacklist performance. Traditional URL reporting was followed by an immediate burst of crawler traffic and a negligible amount of crawler traffic in the hours thereafter. Even though 50% of the phishing websites we reported were successfully retrieved by a crawler, only 20.4% were ultimately blacklisted. The earliest blacklisting occurred 20 minutes after reporting, and coverage stopped growing after approximately four hours. Reporting through CSSR yielded a slower initial speed, but resulted not only in 90.7% coverage within 24 hours, but also a substantially higher amount of crawler traffic, spread over a long period of time, with 47.5% fewer requests being denied by cloaking. The earliest blacklisting occurred 50 minutes after reporting, and coverage matched that of the traditional reporting approach by the five-hour mark. Although we did not repeat these measurements over time, we found the differences in the observed distributions of blacklist speeds to be significant in the Mann-Whitney U test [39], with a p-value below 0.01.

8.7 Crawler Traffic

Between May 2019 and January 2020, the 4,158 URLs in our main deployments received a total of 2.14 million HTTP requests from 41,750 distinct web crawler IPs. An additional 20.50 million requests came from our monitoring infrastructure to check our websites' blacklist status. Our websites remained online for the duration of our deployments (i.e., were not taken down [1]) as we had made our hosting provider aware of the nature of our research. Over time, across all deployments except the third, we also observed a consistent, very slowly increasing level of crawler traffic to our infrastructure, which supports the efficacy of our experimental controls.

55.27% of the crawler requests were successful and returned an *HTTP 200* status code (or *302* for redirection links). The remaining requests returned a *404* status code: 7.56% were denied by cloaking and 37.17% requested nonexistent URLs. Many of the nonexistent URLs represented crawler efforts to scan for phishing kit archives or credential files, which is a common way to fingerprint phishing websites and identify stolen credentials that may linger on the same server [13].

In Figure 9, we show the cumulative distribution of crawler traffic to our websites. We observe that after an initial burst within the first day of deployment, successful traffic remains fairly consistent for the remainder of the deployment. This traffic accounts for crawlers that continuously monitor for the presence of phishing content.

The relative proportion of requests that were denied through cloaking fell over time. The increased crawling effort early on allows crawlers to fingerprint evasion techniques such that future requests are more likely to be successful. We believe that this behavior in part helped blacklists deliver the high coverage we observed, even for websites with combinations of cloaking techniques such as Experiment C.

9 Discussion and Recommendations

Although blacklists are capable of *detecting* the typical evasion techniques which we tested—including cloaked redirection—our tests have shown that these techniques generally both slow speed and reduce coverage. Notable gaps in coverage also remain, particularly on mobile devices. Given attackers' ability to adapt to the ecosystem by leveraging sophisticated evasion strategies, such as those in Experiment F, we believe that evasion remains a key anti-phishing concern.

Defensive Strategy. To the best of our knowledge, systematic longitudinal measurements of anti-phishing defenses

are not currently being performed at the ecosystem level. The PhishTime framework, combined with deployments of targeted experiments, can be used as a defensive strategy to identify gaps in defenses and help address them through security recommendations. Although our work focuses on browser blacklists, the scope of future experiments could also be shifted to evaluate other mitigations (e.g., spam filters). Moreover, the ecosystem analysis could be aimed at areas other than evasion techniques, such as identifying attacker-friendly web hosts or compromised domains [1].

Depending on the objectives of the entity carrying out the experiments, PhishTime can be used to assess aspects of the ecosystem as a whole, or the behavior of a specific entity or mitigation. We believe this is a crucial first step toward achieving consistency in—and perhaps accountability for—anti-phishing and account protection efforts [57] of the many different organizations that phishers impersonate. We have proposed this approach to the APWG; subsequently, efforts are underway to incorporate PhishTime as an ecosystem-level service which can be used to monitor URLs reported to the APWG eCrime exchange and drive future experiments based on this dataset or external reports.

Role of Blacklists. As a supplement to ecosystem defenses, numerous commercial vendors offer phishing website *take-down* services for major brands [2, 45]; such websites are either detected by the vendor's own scanning efforts or reported to the brand. Take-downs are performed via requests sent by the vendor to a hosting provider or domain registrar, are typically reliant on cooperation, and can be subject to delays of several hours or even days [1, 7]. In parallel with our PhishTime experiments, we collaborated with PayPal to measure the take-down speed (for criminal phishing websites) of one major vendor during the same period, and found a median speed of 23.6 hours: considerably slower than the typical speed of blacklists observed in Section 8.2. As blacklists are not subject to the delays inherent to take-downs, we believe that they can better serve as the first line of defense against phishing and may render take-downs unnecessary when their *coverage* is sufficient; this further underscores the benefits of sealing gaps in blacklists' detection of evasive websites.

Reporting Protocols. Given the prevalence of evasive phishing in the wild and the promising performance of CSSR, we believe that the adoption and expansion of evidence-based reporting protocols should be a priority for the ecosystem. In addition to supporting manual reporting by humans, such protocols should be made available to vetted automated anti-phishing systems. A key benefit of such an integration would be if one entity *detects* an evasive phishing website, it could share the parameters used for detection to help other entities (e.g., blacklists) avoid duplication of effort while improving mitigation (e.g., speed and coverage). Moreover, such evidence can be used to support take-down efforts [1] or law enforcement intervention if an initial mitigation, such as blacklisting, proves insufficient. Evidence-based reporting could also help harden systems against abusive, deliberately false reports: such reports could be filtered out based on the evidence itself (e.g., by identifying anomalies within a pool of related reports, rather than solely relying on attributes that are easier to fabricate, such as the bare URL).

Beyond the expansion of enhanced reporting protocols, we believe that standardized methods for reporting phishing *across* the ecosystem—rather than to individual entities—would help improve the ecosystem's collective response. As we observed with single-entity reporting in Experiment E, each anti-phishing entity functioned differently and, thus, affected blacklisting in a different way. Additionally, the drop in coverage we observed during Deployment 3 suggests that the ecosystem may in some cases be fragile. If one anti-phishing entity contributes disproportionately to the mitigation of a particular type of threat, it can become a *choke point*, which, in case of a temporary failure or degradation, could provide an opportunity for phishers to launch a burst of successful attacks. However, strict centralization of reporting could carry privacy or legal concerns; therefore, in a standardized reporting framework, one or more trusted intermediaries could instead serve to optimally route reports.

Certificate Revocation. Throughout our deployments, we monitored the OSCP revocation status [40] of our domains' SSL certificates, which we automatically obtained from Let's Encrypt (a free Certificate Authority with the highest representation among phishing websites in the wild [16]). *None* of the certificates were revoked. In addition, we found that certificates could also be issued for domains that were already blacklisted, as Let's Encrypt had discontinued checking domains in new certificate requests against GSB in early 2019 [33]. Although the role of Certificate Authorities as a mitigation against phishing is subject to debate [16], the ease at which attackers can obtain certificates warrants closer scrutiny.

Mobile Blacklists. Mobile users account for a majority of Internet traffic [14], and prior research has shown that mobile web browsers are particularly prone to phishing attacks [37]. Yet, our findings indicate that the anti-phishing protection in mobile web browsers continues to trail behind that of desktop browsers. The bandwidth used by mobile devices—which may be subject to mobile carrier restrictions—was historically a barrier to desktop-level GSB protection in mobile Chrome and Safari [44]. However, over a Wi-Fi connection (which we used for monitoring), the full blacklist should be checked.

Although our experiments were unable to determine exactly how GSB or Opera decide if a URL blacklisted in the respective desktop browser should also be blacklisted on mobile (e.g., they might rely on manual review or additional classification attributes to determine maliciousness), we observed that the evasive websites we deliberately configured to only be accessible in mobile browsers (i.e., Experiment B) were in fact never blacklisted in these browsers[5]. We therefore

[5]This issue does not apply to mobile Firefox: from version 63, mobile Firefox always checks the full desktop version of the GSB blacklist.

believe that mobile blacklisting represents a key ecosystem vulnerability, and that it should be made consistent to better protect mobile users inevitably targeted by phishers.

Ecosystem Changes. A notable ecosystem development took place in December 2019: in Chrome 79, Google improved the speed of GSB by "up to 30 minutes" [4] by incorporating real-time lookups of phishing URLs for users who opt in. Although not yet enabled by default, this change acknowledges, and seeks to address, the delay to blacklisting speed possible in the default implementation of GSB, which caches and periodically updates a local copy of the URL blacklist. This change also applies to the mobile version of Chrome and may, therefore, help address the aforementioned gaps in blacklisting in mobile Chrome. Due to the timing of the release of this feature, we were not able to evaluate it in our experiments.

9.1 Disclosures

Beyond the disclosures to PayPal and bit.ly discussed in Section 8, after completing our final deployment, we sent a report with our experimental findings to PayPal, Google, Opera, Microsoft, and Apple, with a focus on the sophisticated evasion techniques that we identified and the gaps in blacklisting on mobile devices. All the organizations acknowledged receipt of our report. Google followed up to request details on the JavaScript cloaking, and acknowledged the gap in mobile blacklisting, which it is actively working to address. We later met with Opera who, as a result, incorporated additional ecosystem data sources to improve its discovery of URLs (and ultimately increase blacklist coverage). Moreover, Opera found that these data sources—which enhanced its server-side blacklist—also helped eliminate disparities in mobile blacklist warnings. We described our experimental methodology in each of our disclosures; the former three organizations (which followed up) did not express concerns with it.

9.2 Ethical Considerations

We sought to address a number of potential ethical concerns while conducting this research.

Risk to Human Users. To ensure that our phishing websites could not harm any potential human visitors, we utilized random paths and only distributed the full URLs directly to anti-phishing entities. In the event of form submission, our websites performed no backend processing or logging of POST data; the use of HTTPS ensured that data would not leak in transit.

Infrastructure Usage. We followed the terms of service of all services and APIs used for this research, and we obtained permission from Google to report URLs programmatically to Google Safe Browsing. We informed our hosting provider (Digital Ocean) about our research and obtained permission to leverage server infrastructure accordingly.

Adverse Side-effects. Despite our relatively large sample size for each deployment, we do not believe that the volume of URLs we reported hampered the anti-phishing ecosystem's ability to mitigate real threats. Based on the overall phishing volume per the GSB Transparency Report [25], each of our deployments accounted for less than 1% of all phishing detections made during the same period. We informed PayPal of our experiments to ensure that resources were not wasted on manual investigations of our activity (note that PayPal stated that this knowledge did not influence how it treated the individual phishing URLs we reported). We also obtained permission to use the PayPal brand and promptly disclosed the ecosystem vulnerabilities that we discovered.

9.3 Limitations

Despite the controls discussed in Section 7.4, our experimental findings should be considered alongside certain limitations. We did not modify the appearance of our phishing websites between deployments, and they impersonated a single brand (PayPal). Therefore, our findings may be skewed by detection trends specific to this brand [48]. Possible fingerprinting of the websites' source code over time and the lack of positive reputation of our domains could increase the speed of blacklisting, while our use of randomized, non-suspicious URLs for each website may have reduced it [64]. We believe that our websites still realistically represent phishing in the wild, as attackers extensively re-use phishing kits (some of which share common backends for multiple brands) and also routinely leverage fully randomized URLs [45].

It was not feasible to achieve a one-to-one mapping between our 2,646 unique domains and the 45 hosting IP addresses available to us. To mitigate potential skew from IP reuse, we distributed IP mappings as uniformly as possible within each *batch* of websites. Ultimately, URLs on certain IPs were not significantly more likely to be blacklisted than others: across all experiments, the standard deviation in the distribution of the average GSB blacklist speed by IP was only 3.8 minutes.

When reporting our phishing websites, our goal was to guarantee timely discovery by blacklists. Given the high baseline *speed* and near-perfect *coverage* we observed in Experiment A, we believe that we succeeded in this goal. Nevertheless, unlike real phishers, we did not spam victims or trigger other actions that could lead to detection by blacklists (such as signals from browser-based classifiers [35]). Thus, our reporting methodology may not fully reflect the continuous indicators of abuse observable in the wild; it may also be skewed in favor of GSB: the only blacklist to which we reported directly.

Finally, our experiments were limited in scope to a subset of the different phishing website configurations available to attackers. Additional deployments can naturally be adapted to test other configurations or those that appear in the future. Although the PhishTime framework itself may fail to identify certain attacks that entirely avoid discovery, the use of additional sources of phishing URLs could address this shortcoming.

Fully Automating PhishTime. In our deployment of the PhishTime framework, the website analysis (③) and experiment generation stages (④) were mostly done man-

ually. However, through the addition of a *semantic* phishing classification engine (not only to fingerprint the server-side cloaking of each phishing website, but also to analyze its client-side evasion), end-to-end automation could be achieved for experiments. Manual intervention would then only be needed to evaluate anomalous findings and verify validity.

10 Related Work

To the best of our knowledge, PhishTime is the first systematic methodology for the continuous measurement and enhancement of the protection offered by the anti-phishing ecosystem, and it enabled the first controlled *longitudinal* empirical study of the performance of browser blacklists. Although other controlled studies were previously done, they focused on individual anti-phishing entities and were performed over a short period, which limited the scope of their security recommendations and their ability to validate trends.

The work most similar to ours is that of Oest et al. [44], who proposed the framework for empirically measuring blacklists which we adapted. The authors used the framework to deploy five batches of 396 artificial phishing websites over two weeks to measure five distinct anti-phishing entities' response to websites with different sets of cloaking (similar to our Experiment E). The authors found and disclosed that several cloaking techniques could successfully defeat blacklists, and that due to a bug, mobile GSB browsers saw no blacklisting whatsoever (the latter issue has since been addressed). However, by reporting to just a single entity per batch, the study did not clearly differentiate between blacklist *detection* and *discovery*, and therefore underestimated the ecosystem's *speed*, especially for uncloaked websites. Also, unlike our study, it could not precisely compare the real-world impact of delays in blacklisting caused by different cloaking techniques. Because we guided our experiments by current ecosystem trends (rather than an offline study of cloaking), our experiments more closely emulated real-world attacks and allowed us to evaluate advanced evasion. Consequently, we found new blacklist detection vulnerabilities, long-term inconsistencies, and lingering gaps in blacklisting coverage in mobile browsers. We also found individual cloaking techniques to be a far lesser threat than combinations thereof, which the prior work did not evaluate.

Peng et al. [48] deployed 66 artificial phishing websites over four weeks to investigate how well VirusTotal and its sub-vendors are able to detect phishing content. This study focused on showing that detection models vary greatly across different anti-phishing vendors. These variations help explain the incremental growth that we observed in the coverage of evasive phishing websites across different blacklists.

Other work has indirectly measured the performance of blacklists. Oest et al. [46] analyzed a large sample of phishing traffic to live phishing websites trackable through third-party web requests and found a high degree of sophistication in clusters of large attacks. The authors estimated the average effect of blacklisting across the entire dataset and showed the importance of adequate blacklisting speed by quantifying the potential increase in victims caused by delays; this technique can also help contextualize our experimental findings. Han et al. [26] monitored a honeypot server on which attackers installed phishing kits. Although this approach enables the measurement of attacker and victim interactions with the kits (in addition to the ecosystem's response), the difficulty of controlling variables such as the sample size, deployment time, and website configuration highlights the advantages of our measurement methodology based on artificial websites.

Earlier studies measured the blacklisting of phishing websites *after* they appeared in various feeds [36, 50, 52, 63]. Because feeds have an inherent delay, the resulting measurements of blacklist speed are imprecise. However, they provide insight into the coverage of blacklists across platforms and the characteristics of phishing websites: hence, we adapted this approach in the PhishTime framework and enhanced it with direct reporting to verify blacklists' detection capabilities.

11 Conclusion

We have proposed methodology for systematically evaluating the protection provided by the anti-phishing ecosystem in the long term, with a focus on browser blacklists and phishing reporting protocols. By identifying sophisticated evasion techniques used by phishing websites in the wild and by replicating them in a controlled setting, we were able to identify and help address the gaps that attackers exploit in existing defenses. With a high degree of automation, our approach provides the flexibility to deploy experiments that not only realistically replicate current attacks, but can also be used to proactively test emerging threats or new mitigations.

Anti-phishing defenses used by the ecosystem—including browser blacklists—are highly complex systems. Amid a record volume of phishing attacks [25], these defenses are often capable of quickly responding to phishing websites. Yet, as we observed, even a seemingly small failure in one system component, such as a crawler or reporting channel, may allow certain attacks to succeed and harm users. Analysis of data from our empirical measurement approach can help pinpoint such failures when they have an effect on overall mitigation performance.

Beyond making ecosystem-level recommendations, our approach can benefit organizations impersonated by phishers. Experiments that focus on a specific brand can measure how effectively the ecosystem helps protect that brand's customers, and how well the brand implements its own anti-phishing mitigations, which may otherwise be difficult to measure.

Given the evolution of both phishing attacks and corresponding defenses, we believe that longitudinal measurements of the ecosystem are essential not only for maintaining an understanding of the ecosystem's protection, but also for evaluating new security features as they are released, such that the security of users can be continuously ensured.

Acknowledgments

We would like to thank our shepherd, Paul Pearce, and the anonymous reviewers for their valuable feedback. This material is based upon work supported by the National Science Foundation (NSF) under Grant No. 1703644. This work was also partially supported by PayPal, Inc. and a grant from the Center for Cybersecurity and Digital Forensics at Arizona State University.

References

[1] E. Alowaisheq, P. Wang, S. Alrwais, X. Liao, X. Wang, T. Alowaisheq, X. Mi, S. Tang, and B. Liu. Cracking the wall of confinement: Understanding and analyzing malicious domain take-downs. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2019.

[2] Anti-Phishing Working Group: APWG Trends Report Q3 2019. https://docs.apwg.org/reports/apwg_trends_report_q3_2019.pdf.

[3] T. Berners-Lee, L. Masinter, and M. McCahill. RFC 1738: Uniform resource locators (URL). Technical report, 1994.

[4] Better Password Protections in Chrome. https://blog.google/products/chrome/better-password-protections/.

[5] L. Bilge, E. Kirda, C. Kruegel, and M. Balduzzi. Exposure: Finding malicious domains using passive dns analysis. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2011.

[6] D. Birk, S. Gajek, F. Grobert, and A. R. Sadeghi. Phishing phishers - observing and tracing organized cybercrime. In *Second International Conference on Internet Monitoring and Protection (ICIMP 2007)*, page 3, July 2007.

[7] Bolster: State of phishing and online fraud, 2020. https://bolster.ai/reports.

[8] Browser Market Share. https://netmarketshare.com/browser-market-share.aspx, 2019.

[9] Business E-mail Compromise: The 12 Billion Dollar Scam. https://www.ic3.gov/media/2018/180712.aspx, 2019.

[10] S. Chhabra, A. Aggarwal, F. Benevenuto, and P. Kumaraguru. Phi.sh/$ocial: the phishing landscape through short urls. In *Proceedings of the 8th Annual Collaboration, Electronic messaging, Anti-Abuse and Spam Conference*, pages 92–101. ACM, 2011.

[11] Chrome Suspicious Site Reporter. https://chrome.google.com/webstore/detail/suspicious-site-reporter/jknemblkbdhdcpllfgbfekkdciegfboi?hl=en-US.

[12] J. Cohen. Statistical power analysis for the behavioral sciences. 2nd, 1988.

[13] M. Cova, C. Kruegel, and G. Vigna. There is no free phish: An analysis of "free" and live phishing kits. In *Proceedings of the 2nd Conference on USENIX Workshop on Offensive Technologies*, WOOT, pages 4:1–4:8, Berkeley, CA, USA, 2008.

[14] Desktop vs Mobile vs Tablet Market Share Worldwide. http://gs.statcounter.com/platform-market-share/desktop-mobile-tablet, 2019.

[15] R. Dhamija, J. D. Tygar, and M. Hearst. Why phishing works. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI, pages 581–590, New York, NY, USA, 2006. ACM.

[16] V. Drury and U. Meyer. Certified phishing: taking a look at public key certificates of phishing websites. In *15th Symposium on Usable Privacy and Security (SOUPS'19). USENIX Association, Berkeley, CA, USA*, pages 211–223, 2019.

[17] S. Duman, K. Kalkan-Cakmakci, M. Egele, W. Robertson, and E. Kirda. Emailprofiler: Spearphishing filtering with header and stylometric features of emails. In *Proceedings of the Computer Software and Applications Conference (COMPSAC), 2016 IEEE 40th Annual*, volume 1, pages 408–416. IEEE, 2016.

[18] Economic Impact of Cybercrime- No Slowing Down. https://www.mcafee.com/enterprise/en-us/assets/reports/restricted/rp-economic-impact-cybercrime.pdf, 2019.

[19] J. M. Esparza. Understanding the credential theft lifecycle. *Computer Fraud & Security*, 2019(2):6–9, 2019.

[20] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext transfer protocol – HTTP/1.1, 1999.

[21] Google Docs Phishing Campaign, May 2017. https://www.us-cert.gov/ncas/current-activity/2017/05/04/Google-Docs-Phishing-Campaign.

[22] Google Safe Browsing: Report phishing page. https://safebrowsing.google.com/safebrowsing/report_phish/.

[23] Google Safe Browsing APIs (v4). https://developers.google.com/safe-browsing/v4/.

[24] Google Safe Browsing Submission API. https://cloud.google.com/phishing-protection/docs/quickstart-submission-api, 2019.

[25] Google Safe Browsing Transparency Report. https://transparencyreport.google.com/safe-browsing/overview?hl=en.

[26] X. Han, N. Kheir, and D. Balzarotti. Phisheye: Live monitoring of sandboxed phishing kits. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1402–1413. ACM, 2016.

[27] Y. Han and Y. Shen. Accurate spear phishing campaign attribution and early detection. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, SAC '16, pages 2079–2086, New York, NY, USA, 2016. ACM.

[28] S. Hao, M. Thomas, V. Paxson, N. Feamster, C. Kreibich, C. Grier, and S. Hollenbeck. Understanding the domain registration behavior of spammers. In *Proceedings of the 2013 conference on Internet measurement conference*, pages 63–76. ACM, 2013.

[29] G. Ho, A. Cidon, L. Gavish, M. Schweighauser, V. Paxson, S. Savage, G. M. Voelker, and D. Wagner. Detecting and characterizing lateral phishing at scale. In *Proceedings of the 28th USENIX Security Symposium*, pages 1273–1290, 2019.

[30] L. Invernizzi, K. Thomas, A. Kapravelos, O. Comanescu, J.-M. Picod, and E. Bursztein. Cloak of visibility: Detecting when machines browse a different web. In *Proceedings of the 37th IEEE Symposium on Security and Privacy*, 2016.

[31] M. Konte, R. Perdisci, and N. Feamster. Aswatch: An as reputation system to expose bulletproof hosting ases. *ACM SIGCOMM Computer Communication Review*, 45(4):625–638, 2015.

[32] K. Krombholz, H. Hobel, M. Huber, and E. Weippl. Advanced social engineering attacks. *Journal of Information Security and applications*, 22:113–122, 2015.

[33] Let's Encrypt No Longer Checking Google Safe Browsing. https://letsencrypt.org/2015/10/29/phishing-and-malware.html.

[34] V. G. Li, M. Dunn, P. Pearce, D. McCoy, G. M. Voelker, and S. Savage. Reading the tea leaves: A comparative analysis of threat intelligence. In *Proceedings of the 28th USENIX Security Symposium*, pages 851–867, 2019.

[35] B. Liang, M. Su, W. You, W. Shi, and G. Yang. Cracking classifiers for evasion: A case study on Google's phishing pages filter. In *Proceedings of the 25th International Conference on World Wide Web*, pages 345–356, 2016.

[36] C. Ludl, S. McAllister, E. Kirda, e. H. B. Kruegel, Christopher, and R. Sommer. On the effectiveness of techniques to detect phishing sites. In *Detection of Intrusions and Malware, and Vulnerability Assessment*, pages 20–39. Springer Berlin Heidelberg, 2007.

[37] M. Luo, O. Starov, N. Honarmand, and N. Nikiforakis. Hindsight: Understanding the evolution of UI vulnerabilities in mobile browsers. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 149–162. ACM, 2017.

[38] Manage Privacy: SmartScreen Filter and Resulting Internet Communication. https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/jj618329(v%3Dws.11).

[39] P. E. McKnight and J. Najab. Mann-whitney u test. *The Corsini encyclopedia of psychology*, pages 1–1, 2010.

[40] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams. X. 509 internet public key infrastructure online certificate status protocol-ocsp. Technical report, RFC 2560, 1999.

[41] P. J. Nero, B. Wardman, H. Copes, and G. Warner. Phishing: Crime that pays. In *2011 APWG Symposium on Electronic Crime Research (eCrime)*, pages 1–10. IEEE, 2011.

[42] NetCraft: Report a Malicious Site. https://report.netcraft.com/report.

[43] C. Nykvist, L. Sjöström, J. Gustafsson, and N. Carlsson. Server-side adoption of certificate transparency. In *Proceedings of the International Conference on Passive and Active Network Measurement*, pages 186–199. Springer, 2018.

[44] A. Oest, Y. Safaei, A. Doupé, G. Ahn, B. Wardman, and K. Tyers. Phishfarm: A scalable framework for measuring the effectiveness of evasion techniques against browser phishing blacklists. In *Proceedings of the 2019 IEEE Symposium on Security and Privacy*, pages 764–781, May 2019.

[45] A. Oest, Y. Safaei, A. Doupé, G. Ahn, B. Wardman, and G. Warner. Inside a phisher's mind: Understanding the anti-phishing ecosystem through phishing kit analysis. In *Proceedings of the 2018 APWG Symposium on Electronic Crime Research (eCrime)*, pages 1–12, May 2018.

[46] A. Oest, P. Zhang, B. Wardman, E. Nunes, J. Burgis, A. Zand, K. Thomas, A. Doupé, and G.-J. Ahn. Sunrise to sunset: Analyzing the end-to-end life cycle and effectiveness of phishing attacks at scale. In *Proceedings of the 29th USENIX Security Symposium*, 2020.

[47] Opera Security and Privacy. https://help.opera.com/en/latest/security-and-privacy/#phishingAndMalware, 2019.

[48] P. Peng, L. Yang, L. Song, and G. Wang. Opening the blackbox of virustotal: Analyzing online phishing scan engines. 2019.

[49] PhishTank. https://phishtank.com.

[50] J. P. Randy Abrams, Orlando Barrera. Browser Security Comparative Analysis - Phishing Protection. *NSS Labs*, 2013. https://news.asis.sh/sites/default/files/Browser%20Security%20CAR%202013%20-%20Phishing%20Protection.pdf.

[51] Report a suspicious e-mail or website. https://www.paypal.com/lc/webapps/mpp/security/report-problem, 2019.

[52] S. Sheng, B. Wardman, G. Warner, L. F. Cranor, J. Hong, and C. Zhang. An empirical analysis of phishing blacklists. In *Proceedings of the Sixth Conference on Email and Anti-Spam (CEAS)*, 2009.

[53] SmartScreen: Report a Website. https://feedback.smartscreen.microsoft.com/feedback.aspx?t=0&url=.

[54] A. K. Sood and R. J. Enbody. Crimeware-as-a-service: a survey of commoditized crimeware in the underground market. *International Journal of Critical Infrastructure Protection*, 6(1):28–38, 2013.

[55] B. Stone-Gross, T. Holz, G. Stringhini, and G. Vigna. The underground economy of spam: A botmaster's perspective of coordinating large-scale spam campaigns. *LEET*, 11:4–4, 2011.

[56] Tencent's "two-pronged approach" joins Apple to solve information harassment problem. https://www.qq.com/a/20170614/059855.htm, 2017.

[57] K. Thomas, F. Li, A. Zand, J. Barrett, J. Ranieri, L. Invernizzi, Y. Markov, O. Comanescu, V. Eranti, A. Moscicki, et al. Data breaches, phishing, or malware?: Understanding the risks of stolen credentials. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, pages 1421–1434. ACM, 2017.

[58] K. Thomas, D. McCoy, C. Grier, A. Kolcz, and V. Paxson. Trafficking fraudulent accounts: The role of the underground market in Twitter spam and abuse. In *Proceedings of the 22nd USENIX Security Symposium*, pages 195–210, 2013.

[59] K. Tian, S. T. Jan, H. Hu, D. Yao, and G. Wang. Needle in a haystack: tracking down elite phishing domains in the wild. In *Proceedings of the Internet Measurement Conference 2018*, pages 429–442. ACM, 2018.

[60] E. Ulqinaku, D. Lain, and S. Capkun. 2FA-PP: 2nd factor phishing prevention. In *Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks*, pages 60–70. ACM, 2019.

[61] A. van der Heijden and L. Allodi. Cognitive triaging of phishing attacks. In *Proceedings of the 28th USENIX Security Symposium*, 2019.

[62] Verizon Enterprise Solutions. Data Breach Investigations Report (DBIR). 2019.

[63] N. Virvilis, N. Tsalis, A. Mylonas, and D. Gritzalis. Mobile devices: A phisher's paradise. *2014 11th International Conference on Security and Cryptography (SECRYPT)*, pages 1–9, 2014.

[64] C. Whittaker, B. Ryner, and M. Nazif. Large-scale automatic classification of phishing pages. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2010.

[65] G. Xiang, J. Hong, C. P. Rose, and L. Cranor. Cantina+: A feature-rich machine learning framework for detecting phishing web sites. *ACM Trans. Inf. Syst. Secur.*, Sept. 2011.

[66] W. Xu, F. Zhang, and S. Zhu. The power of obfuscation techniques in malicious javascript code: A measurement study. In *2012 7th International Conference on Malicious and Unwanted Software*, pages 9–16. IEEE, 2012.

[67] Yandex Safe Browsing API. https://tech.yandex.com/safebrowsing/, 2019.

[68] Y. Zhang, J. I. Hong, and L. F. Cranor. Cantina: A content-based approach to detecting phishing web sites. In *Proceedings of the 16th International Conference on World Wide Web*, WWW, pages 639–648, New York, NY, USA, 2007. ACM.